```python
import numpy as np
import matplotlib.pyplot as plt
# defining functions for matrix H with dimension n
def H1(n, E, beta):
    A = []
    for i in range(n):
        B = []
        for j in range(n):
            if j == i:
                B.append(E)
            elif abs(i - j) == 1:
                B.append(beta)
            else:
                B.append(0)
        A.append(B)
    return A

def H2(n, E1, E2, beta):
    A = []
    for i in range(n):
        B = []
        for j in range(n):
            if j == i:
                if (j % 2) == 0:
                    B.append(E1)
                else:
                    B.append(E2)
            elif (abs(j - i)) == 1:
                B.append(beta)
            else:
                B.append(0)
        A.append(B)
    return A

def H3A(n, E, beta):
    A = []
    for i in range(n):
        B = []
        for j in range(n):
            if j == i:
                B.append(E)
            elif abs(i - j) == 1:
                B.append(beta)
            else:
                B.append(0)
        A.append(B)
    A[0][n - 1] = beta
    A[n - 1][0] = beta
    return A

def H3B(n, E1, E2, beta):
    A = []
    for i in range(n):
        B = []
        for j in range(n):
            if j == i:
                if (j % 2) == 0:
                    B.append(E1)
                else:
                    B.append(E2)
            elif (abs(j - i)) == 1:
```

```
            B.append(beta)
        else:
            B.append(0)
        A.append(B)
    A[0][n - 1] = beta
    A[n - 1][0] = beta
    return A

# defining function for calculating energy density function
def D(A):
    import numpy as np
    n = len(A)
    a, v = np.linalg.eig(A)
    # here a is the eigen value and v is the eigen vector
    b = np.sort(a)
    E_min = b[0]
    E_max = b[n - 1]
    delta_E = (E_max - E_min) / n
    print(delta_E)
    E = []
    c = E_min
    E.append(c)
    for i in range(n - 1):
        c += delta_E
        E.append(c)
    Count = []
    for i in E:
        count = 0
        for j in b:
            if j >= i and j < (i + delta_E):
                count += 1
        Count.append(count)
    Density = []
    for i in b:
        k = 0.437288 * pow(abs(i - (-10)), 0.5)
        Density.append(k)
    return E, Count



# Answering the first question
# Ploting D(E) for 5 different values of beta keeping E and N fixed
# I chose E as -15 , N as 500 and beta values as -1,-2,-3,-4,-5
b = [-1,-2,-3,-4,-5]
for i in range(5):
    A=[]
    A=H1(500,-15,b[i])
    x,y=D(A)
    k = x[1] - x[0]
    plt.bar(x, y, width=k)
    plt.title("N=500, E=-15, Beta value: %d" %b[i])
    plt.xlabel("Energy")
    plt.ylabel("D(E)")
    plt.show()
# Ploting D(E) for 6 different values of N keeping E and beta fixed
# I chose E as -15 beta value as -3 and N values as 100,200,500,1000,1500,2000
N = [100,200,500,1000,1500,2000]
for i in range(6):
    A=[]
    A=H1(N[i],-15,-3)
    x,y=D(A)
    k = x[1] - x[0]
```

```
    plt.bar(x, y, width=k)
    plt.title("beta=-3, E=-15, N value: %d" %N[i])
    plt.xlabel("Energy")
    plt.ylabel("D(E)")
    plt.show()



# Answering the second question
# Ploting D(E) for 5 different values of beta keeping E1 and E2 and N fixed
# I chose E1 = -30, E2 = -20 , N as 500 and beta values as -1,-2,-3,-4,-5
b = [-1,-2,-3,-4,-5]
for i in range(5):
    A=[]
    A=H2(500,-30,-20,b[i])
    x,y=D(A)
    k = x[1] - x[0]
    plt.bar(x, y, width=k)
    plt.title("N=500, E1=-30, E2=-20, Beta value: %d" %b[i])
    plt.xlabel("Energy")
    plt.ylabel("D(E)")
    plt.show()
# Ploting D(E) for 6 different values of N keeping E and beta fixed
# I chose E1 = -30, E2 = -20 , beta value as -3 and N values as 100,200,500,1000,1500,2000
N = [100,200,500,1000,1500,2000]
for i in range(6):
    A=[]
    A=H2(N[i],-30,-15,-3)
    x,y=D(A)
    k = x[1] - x[0]
    plt.bar(x, y, width=k)
    plt.title("beta=-3, E1=-30, E2=-20, N value: %d" %N[i])
    plt.xlabel("Energy")
    plt.ylabel("D(E)")
    plt.show()



# Answering the third question(Part-A)
# Ploting D(E) for 5 different values of beta keeping E and N fixed
# I chose E as -15 , N as 500 and beta values as -1,-2,-3,-4,-5
b = [-1,-2,-3,-4,-5]
for i in range(5):
    A=[]
    A=H3A(500,-15,b[i])
    x,y=D(A)
    k = x[1] - x[0]
    plt.bar(x, y, width=k)
    plt.title("N=500, E=-15, Beta value: %d" %b[i])
    plt.xlabel("Energy")
    plt.ylabel("D(E)")
    plt.show()
# Ploting D(E) for 6 different values of N keeping E and beta fixed
# I chose E as -15 beta value as -3 and N values as 100,200,500,1000,1500,2000
N = [100,200,500,1000,1500,2000]
for i in range(6):
    A=[]
    A=H3A(N[i],-15,-3)
    x,y=D(A)
    k = x[1] - x[0]
    plt.bar(x, y, width=k)
    plt.title("beta=-3, E=-15, N value: %d" %N[i])
    plt.xlabel("Energy")
```

```python
    plt.ylabel("D(E)")
    plt.show()
# Answering the third question(Part-B)
# Ploting D(E) for 5 different values of beta keeping E1 and E2 and N fixed
# I chose E1 = -30, E2 = -20 , N as 500 and beta values as -1,-2,-3,-4,-5
b = [-1,-2,-3,-4,-5]
for i in range(5):
    A=[]
    A=H3B(500,-30,-20,b[i])
    x,y=D(A)
    k = x[1] - x[0]
    plt.bar(x, y, width=k)
    plt.title("N=500, E1=-30, E2=-20, Beta value: %d" %b[i])
    plt.xlabel("Energy")
    plt.ylabel("D(E)")
    plt.show()
# Ploting D(E) for 6 different values of N keeping E and beta fixed
# I chose E1 = -30, E2 = -20 , beta value as -3 and N values as 100,200,500,1000,1500,2000
N = [100,200,500,1000,1500,2000]
for i in range(6):
    A=[]
    A=H3B(N[i],-30,-15,-3)
    x,y=D(A)
    k = x[1] - x[0]
    plt.bar(x, y, width=k)
    plt.title("beta=-3, E1=-30, E2=-20, N value: %d" %N[i])
    plt.xlabel("Energy")
    plt.ylabel("D(E)")
    plt.show()
```